REMARKS

Claims 1 to 14 are currently pending in the present application.

It is respectfully submitted that all of the presently pending claims are allowable, and reconsideration of the present application is respectfully requested.

Claims 1 to 8, 10, 11, 13, and 14 were rejected under 35 U.S.C. § 102(b) as anticipated by U.S. Patent No. 5,680,620 ("Ross").

As regards the anticipation rejections of the claims, to reject a claim under 35 U.S.C. § 102, the Office must demonstrate that each and every claim feature is identically described or contained in a single prior art reference. (See *Scripps Clinic & Research Foundation v. Genentech, Inc.*, 18 U.S.P.Q.2d 1001, 1010 (Fed. Cir. 1991)). As explained herein, it is respectfully submitted that the prior Office Action does not meet this standard, for example, as to all of the features of the claims. Still further, not only must each of the claim features be identically described, an anticipatory reference must also enable a person having ordinary skill in the art to practice the claimed subject matter. (See *Akzo, N.V. v. U.S.I.T.C.*, 1 U.S.P.Q.2d 1241, 1245 (Fed. Cir. 1986)).

As further regards the anticipation rejections, to the extent that the Final Office Action may be relying on the inherency doctrine, it is respectfully submitted that to rely on inherency, the Office must provide a "basis in fact and/or technical reasoning to reasonably support the determination that the allegedly inherent characteristic *necessarily* flows from the teachings of the applied art." (See M.P.E.P. § 2112; emphasis in original; and see *Ex parte Levy*, 17 U.S.P.Q.2d 1461, 1464 (Bd. Pat. App. & Int'f. 1990)). Thus, the M.P.E.P. and the case law make clear that simply because a certain result or characteristic may occur in the prior art does not establish the inherency of that result or characteristic.

Claim 1 relates to a computer readable medium having a program, the program performing a method for monitoring an execution of another program that is executable on at least one microprocessor of a micro controller using a debug logic of the micro controller, the method including the features of *causing the debug logic to trigger an exception* upon access to a specific address range during a program execution time, and *causing the debug logic to execute an exception routine* after the exception is triggered during the program execution time, in which *the access to the specific address range includes access to an illegal storage area*, in which *the debug logic and its registers are operated in parallel to the program*

execution time . . . , so as to provide a secure stack check without using the program execution time of the microprocessor, and in which *the debug logic monitors a program run.*

Ross does not identically disclose (or even suggest) all of the features of claim 1. Ross merely refers to a processor having a debug register, in which the processor notifies one program of another program's access to a shared resource. (Ross, col. 2, lines 14 to 24). In this regard, the debug register of Ross is merely a storage block, which is indicated as an integral component of the processor. (Ross, col. 2, line 67, to col. 3, line 2; and col. 4, lines 3 to 36). In short, the debug register of Ross is merely a storage block.

Claim 1 of the present application provides the feature of *causing the debug logic to trigger an exception.* Claim 1 clearly provides that the debug logic -- and not the processor -- triggers an exception. Further, the Specification describes the triggering of an exception as "particularly an interrupt of the program execution." (Specification, p. 8, line 28). In stark contrast, Ross explicitly states that "[w]hen processor 12 detects [the breakpoint] register being set, then *processor 12 generates a device access interrupt.*" (Ross, col. 5, lines 1 to 3 (emphasis added)). Thus, Ross makes plain that its debug register does not trigger an exception. Therefore, Ross does not identically disclose (or even suggest) the feature of *causing the debug logic to trigger an exception*, as provided for in the context of claim 1.

Claim 1 of the present application also provides the feature of *causing the debug logic to execute an exception routine.* Claim 1 clearly provides that the debug logic, and not the processor, executes an exception routine. In stark contrast, Ross explicitly states that "*processor 12 executes a debug interrupt service routine.*" (Ross, col. 5, lines 64 to 65 (emphasis added)). Thus, Ross makes plain that its processor -- and not its debug register -- executes an exception routine. Therefore, Ross does not identically disclose (or even suggest) the feature of *causing the debug logic to execute an exception routine*, as provided for in the context of claim 1.

In addition, claim 1 of the present application provides the feature that *the access to the specific address range includes access to an illegal storage area.* The Specification further describes illegal storage areas as "storage areas which are physically not present or which lie outside a storage area provided," or which are "beyond a preselectable maximum stack size." (Specification, p. 5, lines 15 to 16; and p. 11, lines 1 to 19). In stark contrast, Ross does not disclose any such illegal storage area. Instead, Ross states that "[t]he address which is set as a breakpoint *corresponds to the address which is called when access to the device is desired.*" (Ross, col. 4, lines 42 to 44 (emphasis added)). Thus, the breakpoint

address of Ross is a valid address of a connected device, and plainly is not an illegal storage area which is either physically not present, lies outside a storage area provided, or is beyond a preselectable maximum stack size, as provided for in the present application and in the context of the claimed subject matter.

Therefore, Ross does not identically disclose (or even suggest) the feature that *the access to the specific address range includes access to an illegal storage area*, as provided for in the context of claim 1.

Further, claim 1 provides the features that *the debug logic and its registers are operated in parallel to the program execution time . . . , so as to provide a secure stack check without using the program execution time of the microprocessor.* The present application further describes these features by stating that "the debug logic affects neither the computing performance of the microprocessor nor the program memory," and that "reliable functioning of the fault analysis is provided independently of the computing load of the microprocessor." (Specification, p. 5, lines 22 to 23, and lines 27 to 28).

In stark contrast, Ross explicitly states that "the breakpoint register is monitored by processor 12," and "processor 12 continues to monitor the breakpoint register." (Ross, col. 4, lines 57 to 64). Further, the "processor 12 generates a device access interrupt," and "processor 12 executes a debug interrupt service routine." (Ross, col. 5, lines 2 to 3, and lines 64 to 65). Thus, Ross plainly does not disclose a debug logic operated in parallel to the program execution time, without using the program execution time of the microprocessor.

Therefore, Ross does not identically disclose (or even suggest) the features that *the debug logic and its registers are operated in parallel to the program execution time . . . , so as to provide a secure stack check without using the program execution time of the microprocessor*, as provided for in the context of claim 1.

Moreover, claim 1 of the present application provides the feature that *the debug logic monitors a program run.* The Specification makes plain that the "debug logic . . . is used for monitoring the program for faults." (Specification, p. 5, lines 10 to 11; and p. 8, lines 24 to 26). In stark contrast, Ross explicitly states that "the breakpoint register is monitored by processor 12," and "processor 12 continues to monitor the breakpoint register." (Ross, col. 4, lines 57 to 64). Thus, Ross makes plain that its debug register does not monitor the program for faults. Therefore, Ross does not identically disclose (or even suggest) the feature that *the debug logic monitors a program run*, as provided for in the context of claim 1.

Therefore, Ross does not identically disclose (or even suggest) all of the features of claim 1, including the features of *causing the debug logic to trigger an exception* upon access to a specific address range during a program execution time, and *causing the debug logic to execute an exception routine* after the exception is triggered during the program execution time, in which *the access to the specific address range includes access to an illegal storage area*, in which *the debug logic and its registers are operated in parallel to the program execution time . . . , so as to provide a secure stack check without using the program execution time of the microprocessor*, and in which *the debug logic monitors a program run.*

Accordingly, it is respectfully submitted that claim 1 is allowable for at least these reasons. Claims 2 to 8 depend from and claim 1, and are therefore allowable for at least the same reasons as claim 1.

Claim 10 relates to a control element for a micro controller, and includes features like those of claim 1. Accordingly, it is respectfully submitted that claim 10 is allowable for essentially the same reasons as claim 1, as is its dependent claim 11.

Claim 13 relates to a micro controller, including at least one microprocessor and a debug logic, and includes features like those of claim 1. Accordingly, it is respectfully submitted that claim 13 is allowable for essentially the same reasons as claim 1, as is its dependent claim 14.

Withdrawal of these rejections is therefore respectfully requested.

Claim 9 was rejected under 35 U.S.C. § 103(a) as unpatentable over Ross in view of U.S. Patent No. 6,535,811 ("Rowland et al.").

In rejecting a claim under 35 U.S.C. § 103(a), the Office bears the initial burden of presenting a *prima facie* case of obviousness. In re Rijckaert, 9 F.3d 1531, 1532, 28 U.S.P.Q.2d 1955, 1956 (Fed. Cir. 1993). To establish prima facie obviousness, three criteria must be satisfied. First, there must be some suggestion or motivation to modify or combine reference teachings. In re Fine, 837 F.2d 1071, 5 U.S.P.Q.2d 1596 (Fed. Cir. 1988). This teaching or suggestion to make the claimed combination must be found in the prior art and not based on the application disclosure. In re Vaeck, 947 F.2d 488, 20 U.S.P.Q.2d 1438 (Fed. Cir. 1991). Second, there must be a reasonable expectation of success. In re Merck & Co., Inc., 800 F.2d 1091, 231 U.S.P.Q. 375 (Fed. Cir. 1986). Third, the prior art reference(s) must teach or suggest all of the claim features. In re Royka, 490 F.2d 981, 180 U.S.P.Q. 580 (C.C.P.A. 1974).

As essentially explained above, Ross does not disclose (or even suggest) all of the features of claim 1. Further, it is respectfully submitted that even if it were proper to combine Ross and Rowland, as conclusorily asserted by the Final Office Action (which is not conceded), the secondary Rowland reference does not cure -- and is not asserted to cure -- the critical deficiencies of the Ross reference.

Accordingly, dependent claim 9 is allowable for essentially the same reasons as claim 1, since the Rowland reference does not cure -- and is not asserted to cure -- the critical deficiencies of the primary Ross reference.

Withdrawal of this obviousness rejection is therefore respectfully requested.

Claim 12 was rejected under 35 U.S.C. § 103(a) as unpatentable over Ross, in view of that which the Office Action characterizes as Admitted Prior Art ("APA").

Applicants first note that the Office Action again relies on the BACKGROUND INFORMATION Section of the present application. In this regard, it is noted that while certain published information may represent prior art (namely, any cited patents that are prior art), the information concerning the "debug logic triggering" may represent internal Bosch information.

Regardless of the Office Action's characterization of the "Background Information," (which is not conceded), it is respectfully submitted that the asserted combination does not render unpatentable claim 12 for at least the following reasons. As explained above, Ross does not disclose (or even suggest) all of the features of claim 10. Further, the "Background Information" does not cure -- and is not asserted to cure -- the critical deficiencies of the Ross reference.

Accordingly, dependent claim 12 is allowable for essentially the same reasons as claim 10, since the "Background Information" does not cure -- and is not asserted to cure -- the critical deficiencies of the primary Ross reference.

Withdrawal of this obviousness rejection is therefore respectfully requested.

It is therefore respectfully submitted that claims 1 to 14 are allowable.

Conclusion

It is therefore respectfully submitted that all of the presently pending claims are allowable. It is therefore respectfully requested that the objections and rejections be withdrawn, since all issues raised have been addressed and obviated. An early and favorable action on the merits is respectfully requested.

Respectfully submitted,

Dated: 4/30/2008

By: _____

Gerard A. Messina
Reg. No. 35,952
KENYON & KENYON LLP
One Broadway
New York, New York 10004
(212) 425-7200
CUSTOMER NO. 26646

Reg. No. 33,805
Aaron C. Deslich

1492987